TOOLS AND METHODS FOR COMPUTATIONAL LEXICOLOGY

Roy J. Byrd
Nicoletta Calzolari*
Martin S. Chodorow**
Judith L. Klavans
Mary S. Neff
Omneya A. Rizk***

IBM T. J. Watson Research Center
Yorktown Heights, New York 10598

This paper presents a set of tools and methods for acquiring, manipulating, and analyzing machine-readable dictionaries. We give several detailed examples of the use of these tools and methods for particular analyses. A novel aspect of our work is that it allows the combined processing of multiple machine-readable dictionaries. Our examples describe analyses of data from Webster's Seventh Collegiate Dictionary, the Longman Dictionary of Contemporary English, the Collins bilingual dictionaries, the Collins Thesaurus, and the Zingarelli Italian dictionary. We describe existing facilities and results they have produced as well as planned enhancements to those facilities, particularly in the area of managing associations involving the senses of polysemous words. We show how these enhancements expand the ways in which we can exploit machine-readable dictionaries in the construction of large lexicons for natural language processing systems.

1. INTRODUCTION.

It has become clear that the construction of computer systems that process natural language requires the creation of large computerized lexicons with extensive and accurate syntactic and semantic information about words (see Byrd(1986a), Ingria(1986)). It is also clear that it will be impossible to build these lexicons in the number and sizes required with only the manual labor of individual computer scientists, linguists, or lexicographers. There are too many systems requiring too much information about too many words for the manual approach to succeed.

Fortunately, researchers are beginning to realize that the wealth of information in published dictionaries can be tapped by semi-automatic and fully-automatic methods, in order to help build the computerized lexicons we require. Work reported in Alshawi(1985), Calzolari (1983,1984a,b), Lesk(1986), Michiels(1982), etc., describe various attempts to decipher and extract information in machine-readable versions of published dictionaries (henceforth: MRDs).

The present paper is intended to contribute to that literature by describing the tools and methods used by the Lexical Systems project at IBM Research. Individual tools and their use in that project have been described elsewhere (Chodorow, et al.(1985), Byrd and Chodorow(1985), Byrd, et al.(1986b), Chodorow and Ravin(1987), Neff and Byrd(1987)). This paper differs from previous descriptions of our own and others' work, however, by presenting a framework and a set of general tools and methods that can be re-applied for a variety of lexicological and lexicographical applications. Furthermore, it addresses problems involved in coherently combining information from different MRDs. In particular, we show how we can transfer information from one dictionary to another — even

* Department of Linguistics, University of Pisa and Institute of Computational Linguistics of CNR, Pisa.
** Department of Psychology, Hunter College of CUNY.
***Courant Institute of Mathematical Sciences, NYU.

0362-613X/ 87 /030219-240$03.00

across languages — and correctly associate that information with the word senses to which it applies.

The plan of this paper roughly parallels the logical sequence of problems that face any project in computational lexicology. Section 2 on acquisition and storage describes the initial stages of capturing and organizing dictionary data. Section 3 outlines the tools and methods we use to exploit the contents of machine readable dictionaries. Applications of the tools and methods are described in the final two sections which discuss applied computational lexicography for monolingual and bilingual dictionaries.

2. Acquisition and Storage of Machine-Readable Dictionaries.

2.1. Normalizing Machine-Readable Dictionaries.

Once machine readable dictionary tapes are acquired from the publishers, they usually have to be cleaned up and/or normalized. We have encountered three basic types of publisher tapes: one type, represented by Longman and Webster's Seventh, has some information segmented into separate records labelled with an identifying code and other information implicit in font change codes. A second type, represented by Collins, consists of unsegmented text interspersed with font-change codes. A third type will be the new computerized OED (Tompa (1985)), where each type of information is specifically delimited at the beginning and end, and where there are any number of nesting levels. Much of the work described here was done with data of the second type; however, it is applicable also to the first type, and of course to the third.

In the simplest sense, cleaning up usually means the removal of typesetting codes and other formatting information, or possibly removal of errors (usually done by hand). However, since the typesetting codes (e.g. begin-italic, begin-roman) are clues to the kind of information that follows the codes (for example, part-of-speech information appears in italics in the Collins dictionaries), they cannot simply be washed out. Instead, the raw data undergoes a process of normalization, the goal of which is to identify and label all of the information in the dictionary tapes in such a way that it can be easily accessed without detailed knowledge of the formatting conventions. This is our rationale for the creation of lexical data bases.

We have carried out this normalization process in two stages. The goal of the first stage is to segment the information on the tapes into separate entries, storing the results in a direct access file, where the key is the dictionary headword and the data is simply the body of the entry. Dictionary data is stored in a Dictionary Access Method (DAM) file, from which it can be conveniently retrieved, by headword, for subsequent processing. DAM is completely described in Byrd, et al.(1986b). For this discussion, we note that DAM allows the efficient storage and retrieval of entries for words in alphabetical order and in random order by headword. Having the dictionary in a DAM file allows its entries to be accessed through the WordSmith on-line dictionary system, to be described later.

The goal of the second stage of the normalization process is to segment and identify the data in the entry, creating a second DAM file in lexical data base format. Both stages involve some rather idiosyncratic procedures that are tied to the various ways different publishers code the information on the tapes, particularly since the tapes were not prepared with any consideration of later computational use. However, as will be apparent later, our second-stage normalization is a more general approach that we envision using with a variety of machine-readable dictionary resources. In what follows, we discuss the normalization of the Collins bilingual dictionaries.

The first stage of normalization, picking out the headwords and the associated information, was mostly straightforward: the headwords are preceded by a unique font code. Exceptions to this straightforward process were driven by the principle that the end result of the process should be a direct access dictionary usable by both people and other systems. This involves definition of a file keyword which differs from that of the dictionary headword in two important ways: (1) each keyword must be unique, and (2) there should be a keyword for each spelling of a word, with the important exception of inflected forms predictable by morphological rules, as explained below. The first constraint assures that dictionary lookup for a word will get all of the information associated with that word. The second assures that a lookup operation will find a word where it first expects to find it, and not inside of some other entry.

Three examples will suffice. In adherence to the first constraint, we combined the entries for the noun *quack*^{1} and the verb *quack*^{2} in the English-Italian Collins dictionary. In adherence to the second constraint, we created new entries for alternate spellings of words: the German-English headord *Abtak(e)lung* was made into two entries, *Abtaklung* and *Abtakelung,* the associated information was stored with just one of them, and cross-references to the other spelling were stored with both. Since the notation used for alternate spellings of this sort does not indicate which one is the more common usage, we stored the entry arbitrarily with the shorter form. A third example: where the German-English dictionary stores a number of compound words with the first element of the compound (e.g. *Abend-:* has *-essen, -friede(n) -gesellschaft,* etc.), we created full entries for all of them (e.g. *Abendessen, Abendfriede, Abendgesellschaft,* etc). In the process of reconstituting the compounds, however, we retained information on the location of the break to support possible later work on German noun compounds and to conform to another principle: "do not throw away anything."

An exception to the constraint that there should be separate entries for each spelling of a word, noted before, is that we do not require creation of separate entries for those words which can be derived by morphological rules. For example, from German *Lehrer(in f)*, which is shorthand for *Lehrer* and *Lehrerin f[emale]*, we created only an entry for *Lehrer*, leaving the *(in f)* part for later segmentation. Identification of these exceptions is possible using heuristics and does not depend on having a system of morphological rules for the language of the dictionary. These heuristics are different for each dictionary.

The program which accomplished stage one segmentation was written in a variant of PL/I using recursion to handle the interaction of morphological variants, alternate spellings, and reconstruction of compounds. The results have been used directly for several WordSmith applications.

Once we had the dictionary entries segmented into DAM files, they were ready for the second stage of normalization: identifying and labelling the different types of data in the entries in a standard and consistent way. As we set out to create data bases from the first set of DAM files, it quickly became apparent that the usual data base architecture, characterized by a fixed number of fields for each entry and a fixed amount of space for each, would never work for dictionary entries, because there might be one value for some fields (like spelling), but multiple values for others (like part-of-speech, translation, etc.), or no value at all for still others (e.g. alternate spelling). Moreover, some information, like translation, is associated with values for other information, like part-of-speech. These facts lead to the observation that dictionary entries are best represented as shallow hierarchies of attribute-value pairs. We will return to this observation in the section on "lexical data bases".

We have developed a parser, the goal of which is to convert each dictionary entry into a tree structure where at each node there is an attribute name and a value, which can be a simple value, a list of attribute-value pairs, or an entire subtree. Figure 1 contains the result of applying the parser to the entry for *alert* in the Collins English-Italian dictionary. The attributes in the figure should be familiar to users of bilingual dictionaries; they include hyphenation code ("hyph"), undecoded pronunciations ("pron"), homograph numbers ("hnum"), part-of-speech ("pos"), translations ("spel" under "xlat"), collocations and examples in the source language ("gloss" under "xmp"), and translations of collocations and examples ("tran" under "xmp").

Figure 2 shows how the above-mentioned combination of the multiple entries for *quack* and other words brings about the creation of a "superhom" node. The superhom ("superscript homograph") node serves to separate the multiple entries and preserve the original sense numbers. This is important because cross references in other entries may contain superhomograph numbers as well as sense numbers. Even in entries where there is only one sense, homograph, or superhomograph, these attributes are retained to make access logic consistent with other entries.

```
alert

[entry=
    [hyph=0]
    [pron=>u71<>u11<>u18<1>u26<>u17<t>u72<]
    [hom=
        [hnum=1]
        [pos=adj]
        [sens=
            [xlat=
                [note=acute, wide-awake]
                [spel=sveglio(a)]]
            [xlat=
                [note=mind]
                [spel=pronto(a), agile, vivace]]
            [xlat=
                [note=expression]
                [spel=intelligente]]
            [xlat=
                [note=guard]
                [spel=vigile]]]]
    [hom=
        [hnum=2]
        [pos=n]
        [sens=
            [xlat=
                [spel=allarme]
                [gnd=m]]
            [xmp=
                [gloss=to be on the ~]
                [expl=person]
                [tran=stare all'erta]]
            [xlat=
                [note=troops]
                [spel=essere in stato di allarme]]]]
    [hom=
        [hnum=3]
        [pos=vt]
        [sens=
            [xmp=
                [gloss=to ~ sb (to  sth)]
                 tran=avvisare qn (di qc)
                 tran=avvertire qn (di qc)
            [xmp=
                [gloss=to ~ sb to the dangers of sth]
                [tran=mettere qn in guardia contro qc]]]]]
```

Figure 1. Normalized Collins English-Italian entry for *alert*.

The design of the dictionary entry parser began with a familiar model for parsers of English sentences: a set of rules supported by a parsing engine. Two such models were available to us: a bottom-up, all-paths strategy offered by PLNLP (Langendoen and Barnett(to appear)), and a top-down depth-first approach offered by logic grammars written in Prolog (McCord(1987)). We have versions in PLNLP and VM/Prolog, both of which are basically adequate. The version which supported the work reported here is the Prolog version; however, we are concerned about future problems with the top-down approach when we have to process input that is corrupt. Usually for lexicographical processing, one should not have to worry about large amounts of ill-formed input, but the complex history of some of our tapes forces us to consider possible recovery strategies for text with missing or corrupt font-codes. At that point, it may be necessary to use the bottom-up approach and some recovery strategy analogous to parse-fitting (cf. Jensen, et al.(1983)).

```
quack

[entry=
    [superhom=
        [num=1]
        [hyph=0]
        [pron=>u71<kw>u43<k>u72<]
        [hom=
            [hnum=1]
            [pos=n]
            [sens=
                [xlat=
                    [spel=qua qua]
                    [gnd=m inv]]]]
        [hom=
            [hnum=2]
            [pos=vi]
            [sens=
                [xlat=
                    [spel=fare qua qua]]]]]
    [superhom=
        [num=2]
        [hyph=0]
        [pron=>u71<kw>u43<k>u72<]
        [hom=
            [pos=n]
            [sens=
                [xlat=
                    [note=pej]
                    [spel=ciarlatano/a]]
                [xlat=
                    [note=fam: doctor]
                    [spel=dottore/essa]]]]]]
```

Figure 2. Normalized Collins English-Italian entry for *quack*.

Grammars for dictionary entries and grammars for natural language sentences differ in three important ways: (1) entries can be huge: some are longer than 5000 bytes; (2) there is a different definition of a token, and (3) a dictionary grammar can be to a large extent deterministic. A consequence of (1) is that it takes a large amount of storage to parse an entry. The Prolog version now runs on a 4-megabyte virtual machine under VM/CMS. The motivation for (2) comes from the fact that the entries consist of text interspersed with font-change codes, which are not required to be delimited with blanks. Whereas for ordinary language, the tokens are strings between blanks, for dictionary entries they are the strings of characters between a defined set of delimiters, usually font-change codes, but occasionally also semicolons, periods, and complex character strings. The delimiters are tokens as well. In view of (3), one might well ask why anyone would use a powerful tool like PLNLP or a logic grammar to accomplish something that does not need such powerful support. The answer is that we now have a dictionary parsing engine, a formalism for rules, and an easily modifiable set of rules, all achieved in a shorter development time than would have been accomplished with a conventional programming language. Furthermore, stage two of the acquisition of new dictionaries now requires only writing, testing, and running a new set of grammar rules.

The parsing engine and support code are written in VM/Prolog. The rules, which may be different for each dictionary, are written using a formalism in the spirit of the "definite clause grammars" (DCG's) of Pereira and Warren (1980) and "modular logic grammars" (MLG's) of McCord (1986). A rule compiler translates the rules into Prolog code; this compiler owes much to the groundwork laid by McCord. Significant to McCord's work and necessary for ours here is the distinction between two different kinds of rules, one which produces a structural node (strong non-terminal), and one which produces an intermediate list of nodes (weak non-terminal). The dictionary entry rule compiler sports several new features required by dictionary entry parsing. These include an 'ignore' operator, which causes tokens in the input to be ignored by the automatic structure builder, an 'insert' operator which causes new elements to be inserted, and the efficient implementation of rules consisting entirely of optional elements. This generalized rule-driven parsing engine will be used to help develop entry grammars for all our dictionary resources; a small tool kit for this is already available, and other tools will be added as development progresses.

We illustrate the rule formalism with the following two rules from the grammar for the Collins Italian-English dictionary:

```
hom(nonum) ==>
                (pronunc    | nil ) :
                (pos        | reflex  | nil ) :
                (morph      | nil ) :
                (aux        | nil ) :
                (xref(v)    | nil ) :
                (nil        | gloss) :
                (senslist(*) | nil).
senslist(num) ==> sens(num) : (senslist(num) | nil).
```

The first rule says that an unnumbered homograph consists of a number of optional elements, of which at least one must be present: pronunciation; choice of part-of-speech or verb-reflexive or none; morphological information; selection of perfect tense auxiliary; a certain kind of cross reference (marked with 'v'); a gloss; or a list of senses. The *hom* constituent is declared elsewhere as a strong nonterminal; a node is created for it, with daughter nodes as defined to the right of the arrow.

Prolog backtracking conventions apply; the first successful parse of an entry is used, even if there might be more than one possibility. In the case of most optional constituents, the existence of the constituent is tried before its absence. In the case of the gloss in the first rule, however, the correct result is obtained if the absence of the gloss is tried first.

The second rule says that a list of numbered senses consists of a numbered sense and a numbered sense list (or nothing). The recursive property of this rule allows the assembling of any number of numbered senses. The *senslist* constituent is declared as a weak non-terminal, so that all the sub-constituents defined by the rule will appear as sister nodes at the same level, immediately dominated not by *senslist,* but by its parent, *hom.*

The present version of the grammar has been run on parts of the Collins Italian-English and English-Italian dictionaries with a success rate of about 95%. Some of the entries were quite large. The high success rate is partly due to the consistency and discipline in the formatting of the Collins Italian dictionaries. The rules are still being improved. There are several reasons for the remaining 5% — and, indeed, also for some of the ad hoc devices we have used to successfully parse some entries. Sometimes the primary clues to parsing, the font change codes, can be missing because two discrete data segments are contiguous and appear in the same font. Alternatively, the font changes can actually occur in the middle of a coherent data segment. For example, the translation for English *queen bee* is given as "*ape* f *regina* inv". Here, "f" is the feminine gender marking for *ape,* and "inv" (i.e., "invariable") indicates that the word *regina* remains invariable in the plural form.

In this case, there is grammatical information about each of the pieces of the collocation. In other cases, the first piece of grammatical information attaches to the first word, and the second to the entire collocation. To take a second example, the locution "x *or* y" is often used for phrases in all languages. The problem is that both "x" and "y" can be gapped, or otherwise abbreviated, by deletion of words identical to those found on the other side of the "or". An example is in the translation given for Italian *si sta facendo buio*: "it is growing *o* getting dark" (shorthand for "it is growing dark *o* it is getting dark").

For cases like these, we would ideally like to identify and extract all the information immediately. It would be nice to separate the grammatical information from *ape regina,* in the first case, and to reconstitute the translations *it is growing dark* and *it is getting dark,* in the second. However, we recognize that solving all such problems is a never-ending process and would indefinitely delay the time that any information is available for lexicological analysis. Consequently, we make a practical distinction between the acquisition and analysis phases of our work: information that is not easily decipherable using our entry parsing tools is left in its undeciphered form in the initial lexical data base that we create. Thus, for our examples, we should have the following attributes.

```
[xmp=
    [gloss= queen bee]
    [tran= ape .f. regina .inv.]]
```

```
[xlat=
    [spel= it is growing .o. getting dark]
     . . . ]
```

Later work can be devoted to the detailed analysis of such complex attributes. The results of such analysis will be used to create more refined versions of the initial lexical data bases.

2.2. Lexical Data Bases and the Lexical Query Language.

The lexical data base (LDB) facility is designed to store dictionary information and to provide fast, flexible, and convenient access to it. LDBs differ from DAM files in that the entries are structured into hierarchies of attribute-value pairs, such as those that result from the entry parsing described in the previous section. The lexical query language (LQL) allows users to query, modify, and format those entries. A prototype LDB/LQL system has been built as a testing ground for various concepts and algorithms and is currently in use, as described in sections 4 and 5. The final version of the facility is still under development.

Dictionary entries are typically organized as shallow hierarchies of attribute-value pairs with a variable number of copies of certain nodes at each level. The hierarchical nature of entries stems from the fact that values can be entire subtrees. Thus, after processing by the dictionary entry parser described in the previous section, the entry for *quack* from the Collins English-Italian dictionary can be stored in an LDB as shown in Figure 3. In this hierarchy, the "hdw" terminal node contains the headword of the entry. Dictionary workers are familiar with the notion of hierarchically structured entries of this type.

It is important to distinguish between the type of hierarchies found in dictionary entries and those, for example, that characterize the syntactic structure of English sentences. Both types of hierarchy are, in principle, of unbounded size. Dictionary entries achieve their unboundedness by iteration: a bilingual dictionary entry may have any number of homographs, which may, in turn, have an arbitrary number of senses, and so forth. Syntax trees achieve their unboundedness both by iteration (e.g., coordination) and by recursion (e.g., self-embedding). The observation that dictionary entry hierarchies may only iterate is useful for defining lexical data bases. All entries in an LDB can be characterized by a "design" which is stored once for the data base. The design is a single hierarchy which serves as a "grammar" for any instance of an entry by naming the possible parent, siblings, and children of each node type that may occur. Actual LDBs are created by storing hierarchically formatted entries in a DAM file with the design stored as control information. Each entry is stored with its headword as key. Alternate access paths can be established by building indexes on other attributes.

```
entry
+-hdw quack
|
+-superhom
| +-hum 1
| +-hyph 0.
| |
| +-pron >u71<kw>u43<k>u72<
| |
| +-hom
| | +-hnum 1
| | +-pos n
| | |
| | +-sens
| |   |
| |   +-xlat
| |     +-spel qua qua
| |     +-gnd m inv
| |
| +-hom
|   +-hnum 2
|   +-pos vi
|   |
|   +-sens
|     |
|     +-xlat
|       +-spel fare qua qua
|
+-superhom
  +-hum 2
  +-hyph 0.
  |
  +-pron >u71<kw>u43<k>u72<
  |
  +-hom
    +-pos n
    |
    +-sens
      |
      +-xlat
      | +-note pej
      | +-spel ciarlatano/a
      |
      +-xlat
        +-note fam: doctor
        +-spel dottore/essa
```

Figure 3. LDB representation for the Collins English-Italian entry for *quack*.

LQL allows the user to specify conditions on the attributes of LDB entries. Only those entries which satisfy all conditions become part of the query answer. Further, the user specifies which attributes of the successful entries remain in the answer and what their output format will be. The query is stated as entries in the nodes of a two-dimensional representation of an LDB's design. The query syntax is reminiscent of the syntax of the Query-by-Example (QBE) data base query language (Zloof(1974)). Example-elements, denoted by a leading underscore, are used to relate values of attributes in a query tree to conditions in a "condition box", to display positions in an "output box", and to attribute values in other dictionary entry trees.

Part (a) of Figure 4 shows a query which will list all words which are both nouns and verbs in English together with their translations in the Collins English-Italian dictionary. The condition on the noun part-of-speech attribute is simple (it must be "n" for this data base) and is entered directly in the tree. The condition on the verb part of speech is more complex, and the example-element _vpos is used to relate those attribute values to the condition box, where they are tested for equality with either "vi" or "vt". The example-elements _word, _ntran, and _vtran are used to map answers from the hierarchy into the output box where their eventual output format is schematically represented. Part (b) shows sample entries from the answer to this query when applied to the Collins English-Italian dictionary. Such a query might be useful, for example, in a study of the relation between homography in English and derivational morphology in Italian.

```
(a) Query

entry
+-hdw _word                       +--------- OUTPUT --------+
|                                 |                         |
+-superhom                        | WORD: _word             |
  |                               |   NOUNS          VERBS  |
  +-hom                           |   _ntran        _vtran  |
  | +-pos n                       |                         |
  | |                             |                         |
  | +-sens                        |                         |
  |   |                           +-------------------------+
  |   +-xlat
  |     +-spel _ntran             +------ CONDITIONS -------+
  |                               |                         |
  +-hom                           | _vpos = vi | _vpos = vt |
    +-pos _vpos                   |                         |
    |                             |                         |
    +-sens                        +-------------------------+
      |
      +-xlat
        +-spel _vtran

(b) Answer:

WORD: force
   NOUNS          VERBS
   forza          forzare
                  costringere

WORD: quack
   NOUNS          VERBS
   qua qua        fare qua qua

WORD: scream
   NOUNS          VERBS
   grido          gridare
   strillo        urlare
   urlo
```

Figure 4. An LQL query.

LQL conditions specify tests to be performed on the values of attributes in the data tree to which nodes in the query tree may be mapped during query processing.

Terminal nodes may be tested using a variety of string and arithmetic operations. The current prototype implementation includes the built-in functions of the REXX programming language (IBM (1984)). Non-terminal nodes may only be tested for equality or inequality with other nodes of the same type. All nodes may have aggregate functions (e.g., count, maximum, minimum, etc.) applied to them and the results may either be tested in conditions or be output as part of the query answer. Nodes may also be tested for a null value (i.e., non-occurrence in a particular entry).

As in the query tree itself, the output box may contain both constants and example-elements. The constants define boilerplate material used to label the variable instantiations of the example-elements, as in Figure 4. Such labelling is perhaps more useful when dictionary entries are presented one at a time, as in the WordSmith on-line dictionary system (Neff and Byrd(1987)).

A realistic data base system must provide facilities for creating and maintaining data bases. LDB creation is usually a bulk operation and has been discussed in the previous section. LDB maintenance, on the other hand, can benefit from the flexibility provided by combining a powerful query processor with the capability to insert, delete, and update parts of the LDB entries. LQL offers this flexibility by providing the operators "i." (for "insert"), "d." (for "delete), and "u." (for "update"). These are also familiar Query-by-Example concepts and are described in a relational context in IBM(1978), and in the context of LDBs in Neff, et al.(1988). LDB modification will not be discussed further in this paper.

We envision that LQL will be available in multiple environments. Naturally as a stand-alone processor, it can be used to specify batch query and modification operations against entire lexical data bases. This processor and the flexibility with which it can produce results tailored to a specific problem make LQL a valuable tool for lexicological research. We also plan to use LQL as an entry formatting and filtering mechanism for the WordSmith on-line dictionary system.

3. Tools and Methodologies.

3.1. Tools.

In this subsection, we give an alphabetical listing of the major computational tools, other than the dictionary entry parser and LDB/LQL, at our disposal for doing lexicological analysis of our MRDs. The tools are described in more detail in the sections of the paper where their use in a certain application is presented. Further detail is available in the references. Minor tools used for individual applications are not shown here.

The tools are in the form of a variety of programs written by different members of the Lexical Systems project. The programming languages used include IBM's EXEC2, REXX, PL/I, and VM/Prolog languages. What makes them coherent are the fact that we use the Dictionary Access Method and its utilities as a common interface to the data and that we share a common set of methods and goals when we apply them.

DAM. The Dictionary Access Method (Byrd, et al.(1986b)) provides random and sequential access to dictionary entries associated with words which serve as search keys. The sequential access can be modified to observe "correct" alphabetical orders in any of several languages. An associated set of utilities makes the management of DAM files (including creation, updating, and compression) especially convenient.

DICTUTIL. DICTUTIL is a menu-driven interface to the prototype implementation of the lexical data base system. It allows the analyst to create LDBs using output from the parsing of Collins bilingual dictionary entries and to execute simple queries against those LDBs. The output from a DICTUTIL query is stored as an index to the original LDB (i.e., as a set of value-headword pairs).

Filtering. Filtering, described in Chodorow, et al.(1985), is a method for expanding a set of words each of which has a given property. It uses the hypernym relationship, produced by Head Finding, or the synonym relationship to find new words that are related only to words already in the set. Such new words are added to the set and the process repeats until convergence. See section 3.2, below.

Head Finding. Head finding, also described in Chodorow, et al.(1985) and in Calzolari(1984a), is a method for automatically discovering the hypernym (or genus term) for a word by analyzing its dictionary definition. It is based on the observation that, at least for nouns and verbs, the syntactic head(s) of a definition is(are) usually the hypernym(s) of the word being defined. See section 3.2, below.

Matrix Building. The Matrix Building program takes ordered pairs of words that bear a given relationship to one another (e.g., X is a synonym of Y) and constructs a matrix in which each X word is represented by a row, each Y word is represented by a column, and the relationship between them is indicated in the cell that is formed by the XY intersection. Synonym matrices are used for clustering synonyms into senses. Translation matrices (where X is a translation of Y) are used for transferring and analyzing sense information using bilingual dictionaries. See sections 5.1 and 5.2, below.

Sprouting. Sprouting, described in Chodorow, et al.(1985), is a method for exploring the hypernym, hyponym, and synonym relations. It finds words related to some initial seed by computing the transitive closure over the relation (i.e., "sprouting a tree") beginning from the seed. See 4.1, below.

TUPLES. TUPLES (Byrd(1986b)) is a text analysis system for finding frequent words and phrases in ordinary text. It uses UDICT to provide lemma normalization and base normalization as it compares strings from the text. TUPLES is used in the analysis of dictionary definitions for locating and counting defining formulae and presenting them in a key-word-in-context format.

UDICT. UDICT (Byrd(1986a)) is a computerized lexicon system. It supplies a rich variety of lexical information to natural language processing applications. Of particular interest for lexicography is the fact that UDICT performs morphological analysis of words having inflectional and derivational affixes as well as for compound words and multi-word entries. It also has a facility for producing correctly inflected form of English words. UDICT is a component of TUPLES and WordSmith.

WordSmith. WordSmith is an on-line dictionary system, described in Neff and Byrd(1987), which allows flexible access to dictionaries stored as DAM files and lexical data bases. Although not explicitly mentioned in the remainder of the paper, it is an important reference tool for all of our research. Among its capabilities, WordSmith provides access to:

- definitions, synonyms, and etymologies from Webster's Seventh (Merriam(1963)),
- pronunciations from Webster's Seventh and rhymes based on them,
- definitions and grammatical information from LDOCE (Longman(1978)),
- synonyms from the Collins Thesaurus (Collins(1984)),
- entries from the Collins bilingual dictionaries for English/Italian, English/French, English/Spanish, and English/German (Collins(1971, 1978, 1980, 1981)).

3.2. Methodologies.

At the beginning of our work, we dealt with words as undifferentiated wholes. That is, we only stored lexical information for a word when it applied to all of the word's senses for a given part of speech. Even though this practice made us somewhat uncomfortable, it allowed us to make significant progress in dictionary analysis. The primary reason, as shown in Figure 5, is that most words in the dictionaries we analyze are monosemous; that is, they have just one sense. Of the approximately 60,000 headwords in Webster's Seventh having fewer than ten senses, almost two-thirds of them (over 38,000) are monosemous. Further analysis, not given here, shows that, even among words that have multiple parts-of-speech, each part-of-speech is also typically monosemous. This further increases the proportion of the dictionary entries that we can handle with our earliest techniques. Analysis of Zingarelli(1971) reveals that the facts for Italian are analogous to those for English: over all parts-of-speech, 61.7 percent of the Italian entries were monosemous; 38.3 percent were polysemous. Further details of the Italian study are presented in Calzolari(1980).

These statistics explain why MRD-based methodologies such as head-finding, sprouting, and filtering are as productive as they are, even though they don't handle polysemy correctly. They are described in section 3.2.1. Despite these encouraging statistics, however, it is ultimately necessary to be able to deal correctly with the multiple senses of polysemous words. The need is especially acute because the most frequently used words tend to also be the most polysemous. Two specific problems associated with polysemy in MRDs are the mapping problem and the addenda problem. Section 3.3 describes these problems and offers solutions for them.

senses/word	percent	# words
1	64.1%	38446
2	24.1%	14415
3	6.9%	4117
4	2.6%	1557
5	1.2%	699
6	0.6%	347
7	0.3%	203
8	0.2%	107
9	0.0%	43

		59934 Total words

Figure 5. Number of senses per word in Webster's Seventh.

3.2.1. Head Finding, Sprouting, and Filtering.

One goal of our research is to extract semantic and syntactic information from standard dictionary definitions for use in constructing lexicons for natural language processing systems. Dictionaries are rich sources of detailed information, but in order to use the information for natural language processing, it must be organized systematically.

Amsler(1980) demonstrates that additional structure can be imposed upon a dictionary by making certain assumptions about the ways in which definitions are constructed. Foremost among these assumptions is that definitions consist of one or more "genus" terms, which identify superordinate categories of the defined word, and "differentia" which distinguish this instance of the superordinate categories from other instances. By manually extracting and disambiguating genus terms for a pocket dictionary, Amsler demonstrated the feasibility of generating semantic hierarchies.

It has been our goal to automate the genus extraction and disambiguation processes so that hierarchies can be generated from full-sized dictionaries, such as *Webster's Seventh New Collegiate Dictionary*. These efforts are described in detail in Chodorow, et al. (1985). They begin with Head Finding.

In the definition of *car* as "a vehicle moving on wheels", the word *vehicle* serves as the genus term, while "moving on wheels" differentiates cars from some other types of vehicles. Taken as a group, all of the word/genus pairs contained in a normal dictionary for words of a given part-of-speech form what Amsler(1980) calls a "tangled hierarchy". In this hierarchy, each word constitutes a node whose subordinate nodes are words for which it serves as a genus term. The words at those subordinate nodes are called the word's "hyponyms". Similarly, the words at the superordinate nodes for a given word are the genus terms for the various sense definitions of that word. These are called the given word's "hypernyms". Because words are polysemous any word may have multiple hypernyms; hence the hierarchy is "tangled".

Our automated approach to finding the genus terms in definitions is based on the observation that the genus term for verb and noun definitions is typically, though not always, the syntactic head of the defining phrase. This reduces the task to that of finding the heads of verb phrases and noun phrases. The syntax of the verb phrase used in verb definitions makes it possible to always locate its head with a simple heuristic: the head is the single verb following the word *to*, or if there is a conjunction of verbs following *to*, then they are all heads.

Head finding is much more complex for noun definitions because of their greater variety, but we have taken advantage of the fact that dictionary definitions are written in a special and predictable style to develop a heuristic procedure for finding the head. The procedure can be described briefly as follows. First the substring which must contain the head is found. This substring is bounded on the left by a word which obligatorily appears in prenominal position: *a, an, the, its, two, three, . . ., twelve, first, second,* and so forth. It is bounded on the right by a word or sequence that can only appear in postnominal position, such as a relative pronoun, a preposition, or a present participle following a noun.

Once the substring is isolated, the search for the head begins. Typically, but not always, it is the rightmost noun in the substring. If however, the substring contains a conjunction, each conjunct is processed separately, and multiple heads may result. If the word found belongs to a small class of "empty heads" (words like *one, any, kind, class, manner, family, race, group, complex,* etc.) and is followed by *of*, then the string following *of* is reprocessed in an effort to locate additional heads. The accuracy of head-finding for nouns was approximately 98 percent, based on a random sample of the output.

The word defined and the head of its definition are the raw materials for two semi-automatic processes which make explicit an important part of the lexical organization and featural information that are held implicit in dictionaries. The first of these is sprouting.

Sprouting, which derives its name from the action of growing a semantic tree from a specified root, uses the results of head-finding organized into a "hyponym index", in which each word that was used as a genus term is associated with all of its hyponyms. Thus, *vehicle* would have an entry in the index which reads (in part):

vehicle: ambulance . . . bicycle . . . car . . .
tanker . . .

For a given part-of-speech, the hyponym index needs to be built only once.

When invoking the SPROUT program, the user selects a root from which a semantic tree is to be grown. The system then computes the transitive closure over the hyponym index, beginning at the chosen root. In effect, for each new word (including the root), all of its hyponyms are added to the tree. This operation is applied recursively, until no further new words are found.

The interactiveness of the sprouting process results from the fact that the user is consulted for each new word. If the user decides that the word does not belong to the tree being grown, it can be pruned. These pruning decisions result in the disambiguation of the tree, though clearly not by means of an automatic process.

The output of a sprouting session, then, is a disambiguated tree extracted from the tangled hierarchy represented by the hyponym index. The words it contains all have at least one sense which bears the property for which the root was originally selected. It is important to note that any serious use of sprouting to locate all words bearing a particular semantic feature must involve the careful selection and use of several roots, because of the variety of genus terms employed by the Webster's lexicographers. This will be quite obvious in the case studies cited below.

Filtering, like sprouting, results in lists of words bearing a certain property (e.g., [+human]). Unlike sprouting, however, filtering only picks up words all of whose senses have the property. It is based on a hypernym index (the inversion of the hyponym index), in which each word is listed with its hypernyms, as in the example given here:

vehicle: agent equipment means medium

The filtering process begins with a "seed filter" consisting of an initial set of words all of whose senses bear some required property. The seed filter may be obtained in any manner that is convenient. Several approaches to setting up a seed filter are discussed in the case studies that follow. Given the filter, the system uses it to evaluate all of the words in the hypernym index. Any words, all of whose hypernyms are already in the filter, become candidates for inclusion in the filter during the next pass. The user is consulted for each candidate, and may accept or reject it. Finally, all accepted words are added to the filter, and the process is repeated until it converges.

Like sprouting, filtering produces a list of words having some desired property. In the case of filtering, however, the resulting words have the property in all of their senses.

3.3. Multi-dictionary Lexicology.

When lexicographers create dictionaries, they identify varying numbers of senses for words based on such things as dictionary size, editorial policy, available citations, and intended audience. Computer scientists who create or add to a dictionary for supporting natural language processing applications have other criteria which guide the sense distinctions to be made. It is not to be expected that different MRDs will agree on the numbers or orders of senses for the words they contain. The disagreement, however, raises problems for computational lexicology, two of which are discussed in this section.

3.3.1. THE MAPPING PROBLEM.

When processing multiple MRDs, one encounters what we have termed the "mapping problem". The problem is to map the senses given for a word in one dictionary onto the senses given for the same word in another dictionary. We see a mapping as a symmetric binary relation between (word senses in) two dictionaries for the same language.

A solution to the mapping problem is necessary in order for us to combine information in different published dictionaries. For example we might want to relate grammatical information from the Longman Dictionary of Contemporary English with definitions from Webster's Seventh. Having correct mappings will also allow us to transfer information from one dictionary to another when we are building new computerized dictionaries such as UDICT. We note that some computerized dictionaries will have artificially created sets of senses — possibly motivated by discriminations required by the application being served. Our mapping strategy must be capable of handling these also.

Figure 6 contains a general picture of the mapping relation among 3 different dictionaries (Webster's Seventh, Collins English-Italian, and Italian monolingual) together with evidence for establishing a specific mapping.

In the figure, the English word E1 has three senses in Webster's Seventh, two known to refer to humans and one to an instrument. The Collins English-Italian dictionary has two senses for E1; with two Italian words I1 and I2 given as translations of the respective senses. Finally a monolingual Italian dictionary contains the information that I1 is human and that I2 is an instrument. Using techniques and representations to be described below, we can establish the mapping shown between Webster's Seventh and Collins English-Italian; the first two senses in Webster's map to the first sense in Collins and the third in Webster's maps to the second in Collins.

Solving the mapping problem involves two kinds of activities. First, a representation must be chosen for storing the mappings as they emerge from various procedures intended to find them. This decision is merely one of choosing permanent file structure for the binary relations. (Note that we reject the alternative of discovering mappings dynamically when they are needed because (1) no single procedure can be devised to find them all and (2) most procedures for finding any mappings would be too expensive for dynamic application.) In the second activity, we must devise and run procedures for obtaining the detailed mappings. In general, multiple procedures will be required in order to fully cover the words in the two dictionaries to be mapped.

Any file system that supports random access of information keyed by words will be suitable for storing the mappings. DAM will be used for the mapping relations built at IBM Research. In the DAM files, headwords from the two dictionaries being mapped will serve as keys.

The information stored for each headword can be a set of binary mappings. For example, we give here the mapping for the word E1 in Webster's Seventh and Collins English-Italian shown in Figure 6:

1,2:1;3:2

This can be read as: Webster's senses 1 and 2 map onto Collins's sense 1; Webster's sense 3 maps onto

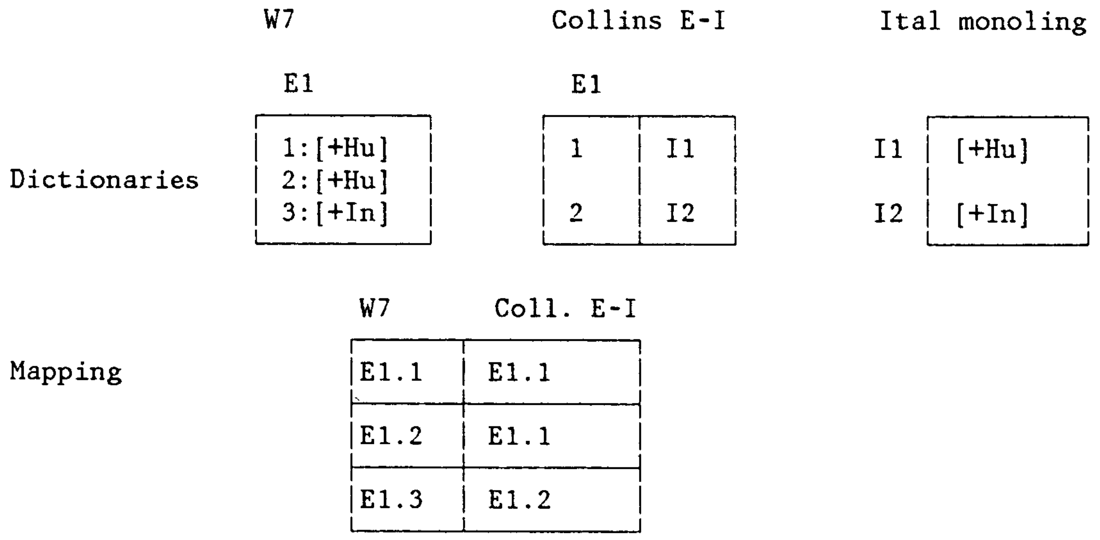

Figure 6. Mapping from Webster's Seventh to Collins English-Italian.

Collins's sense 2. Here the binary mappings are separated by semicolons. Each mapping consists of two lists of senses separated by colons. The leftmost list is for Webster's Seventh, the rightmost for Collins English-Italian. Since the mappings are symmetric, the assignment of left or right to a particular dictionary is arbitrary. Each list, in turn, is a comma-delimited set of sense numbers. A list may be null if one dictionary excludes a sense contained in the other. (It is important to note, however, that the success of the mapping strategy depends on the MRDs involved having fairly complete coverage of the language. Arbitrary omission of word senses will, in general, lead to incorrect results.)

With respect to the number of non-null word senses in the two entries for a given word, there are four possible types of mappings, as shown in the table in Figure 7.

Type	# senses on left	# senses on right	entry in mapping file
1	1	1	1:1
2	1	>1	1:1,2(,...)
3	>1	1	1,2(,...):1
4	>1	>1	varied

Figure 7. Types of mappings.

We can imagine very simple strategies for populating the mapping file for mappings of types 1,2 and 3. We simply build the mapping entry shown in the fourth column of the figure. Given our previous observation that most entries in a typical dictionary are monosemous, these strategies will account for a significant portion of the mapping file entries.

Mappings of type 4 will require more involved strategies. As an example of one such strategy, consider the example in Figure 6. If I1 is [+human] and is the only translation given for the first sense of E1 in Collins English-Italian, then the first two senses of E1 in Webster's may be mapped to the first sense of E1 in Collins. Likewise, if I2 is [+instr] and is the only translation given for the second sense of E1 in Collins, then we may assume that the third sense of E1 in Webster's (which is [+instr]) can be mapped to the second sense of E1 in Collins. A procedure based on these observations will yield the mapping shown before, namely:

1,2:1;3:2

Note that the procedure will include generalizations of some aspects of the observations made above. For example, if Collins gives multiple translations for a single sense of a word, and if all are [+human] that is as good as having a single translation. Likewise, if a single translation is polysemous in Italian but all senses bear the relevant feature, then the observations still hold. Further generalizations should yield a useful procedure for instantiating (part of) the desired mapping.

Many such procedures, possibly augmented by some manual labor, will be required to build an entire mapping. In particular, we expect that for highly polysemous and common words (such as *run, go, turn, thing, make, be,* etc., in English) the mappings will have to be done by hand. Once obtained, however, the mapping will be permanent and can be used for a variety of purposes, some of which will be discussed in sections 4 and 5 of this paper.

3.3.2. The Addenda Problem.

Sense mapping is not the only aspect of work with computerized dictionaries where the ability to identify word senses is critical. Activities which develop dictionary information—either by dictionary and text analysis or by transferring the information from other dictionaries—must ultimately have word senses as targets for that information. The information to be associated with the target dictionary is generally represented either as features, or as relations (i.e., attribute-value pairs).

The dictionary to which the information is to be attached may either be a published dictionary, like Webster's Seventh, or a dictionary intended for computer use, such as UDICT. In the former case, the senses are those established by the lexicographers. In the latter case, the senses are those which meet the needs of the application(s) which the dictionary serves.

We will also need to decide whether the new information should actually be added to the existing file for the target dictionary or, rather, whether it should be kept in a separate "addendum" to the base file, leaving the base file unchanged. Ignoring, for the moment, the first possibility, we propose the following strategy for adding new information as an addendum to an existing base dictionary.

The information must be addressable by word and sense number, where the sense numbers are those given in the base dictionary. However, the need to access information by sense number, independently of the word, will never arise. Consequently, a DAM file in which headwords from the base file are keys, and in which sense numbers are used to organize the data records, is a suitable implementation. In particular, the file can be organized as an LDB whose entries have the following structure:

```
entry
+-hdw
|
+-superhom
  |
  +-hom
  |
  +-sens
    +-snum
    - . . . (new information) . . .
```

To take a concrete example, suppose that analysis techniques described in this paper have been used to determine that the first sense of the noun *mother* in Webster's Seventh is [+animate] and takes [+animate] nouns as typical objects (*the mother of my puppies* is grammatical but **the mother of my buggy* is not). Further analysis will show that the third sense of *mother* (''maternal tenderness or affection'') is [+abstract] and [+mass]. The LDB entry for *mother* in the addenda file for Webster's Seventh will have the following structure:

```
entry
+-hdw mother
|
+-superhom
  |
  +-hom
    |
    +-sens
    | +-snum 1
    | +-animate yes
    | +-typicalobj animate
    |
    +-sens
      +-snum 3
      +-abstract yes
      +-mass yes
```

With join operations in LQL, we will have a flexible means of querying, formatting, and otherwise combining the new information with that from the base dictionary.

4. Applied Computational Lexicology: Monolingual Dictionaries.

As we pursue our goal of building computerized dictionaries containing extensive lexical information, we use our data bases, tools, and methods to perform a variety of analyses. This section and section 5 present samples of those analyses. The variety will be evident in the different types of results sought and methods used. It is also reflected in the fact that some of these analyses have been completed, whereas others are still in progress.

4.1. Extracting Syntactic and Semantic Information.

We have established a list of semantic and syntactic information which we intend to obtain for the purpose of building dictionaries suitable for natural language processing. For nouns, we have chosen information which is primarily motivated by selectional restrictions, such as [+human]/[+nonhuman], [+male]/[+female], [+abstract]/[+concrete]. Also for nouns, we are identifying attributes such as ''made_of'' and ''set_of'', and features such as [+unit] (explained below). For verbs, we are currently extracting information such as [+thatcomp] (takes a *that* complementizer), [+active]/ [+stative], [+reflexive], [+reciprocal] and [selects for a subject of X type]. Note that some of this information can be represented in terms of binary features, whereas other information should be represented as attribute-value pairs.

There appears to be independent consensus in the computational literature for choosing information such as this. For example, Ingria (1986) refers to a set of lexical features necessary for the construction of large lexicons both for text analysis and generation systems. McCord(1987) refers to the features [+human]/ [+nonhuman], [+male]/[+female], and [+animal]/ [+nonanimal]. For McCord, semantic constraints can be exercised by operations on hierarchies of features with implicational relations. Dahlgren and McDowell(1986) present a system for reasoning about natural language texts. Among the primitive concepts in their system are ones we too have identified, such as PROCESS, ACTIVITY, MOTION, ANIMATE, INANIMATE, INDIVIDUAL, COLLECTIVE, and so on.

We believe that some of the same lexical information may be required for different languages. So far we have preliminarily examined Italian and English; the information we have identified for each language independently turns out to be of mutual interest and utility. This suggests to us that as we examine other languages, we might find a core set of syntactic and semantic attributes of universal linguistic interest which should be represented in the computational lexicon across languages.

4.1.1. [+male] and [+female] Nouns.

Marking nouns with a feature reflecting semantic gender in English is motivated by both syntactic and semantic facts. Syntactically, processes like pronominalization are dependent on gender in English. Semantically, certain verbs, such as *impregnate,* select for [+male] or [+female] arguments. The criteria for defining [+male] and [+female] are relatively straightforward. We chose the option of two features because this is not a binary distinction, e.g. *happiness* is both [-male] and [-female], whereas *author* could be either [+male] or [+female].

We used three methodologies to extract nouns that are candidates for marking with the gender features. The initial list was extracted from Longman using the semantic codes which indicate "male human", "female human", "male animal", and "female animal". We then ran sprouting and filtering to enlarge the lists. The preliminary lists for the [+male] and [+female] features using sprouting and filtering were processed by Barbara Ann Kipfer (see Kipfer(1985)). The lists so obtained consisted of 732 entries for the [+male] feature and 519 for the [+female] feature.

In addition to filtering, we searched entries containing the key words *man, boy, husband, son,* and *male* for the candidate [+male] words, and *woman, girl, wife, daughter,* and *female* for potential [+female] words. The simple key-word search technique turned up many items, particularly from the domain of botany, that were not necessarily [+male] or [+female]. This is not unexpected, and indeed confirms the motivation for developing an intelligent system such as filtering, rather than a less clever system such as the simple key-word search. However, the key-word search provided us with items that filtering did not catch. In particular, multiword entries, such as *man Friday* or *office girl* were not picked up by filtering, but were picked up by the key-word search. Also problematic were noun compounds within a definition, such as:

AVIATRIX (n) a woman aviator — called also aviatress
CHURCHWOMAN (n) a woman adherent of a church

The heads of these definitions are *aviator* or *adherent,* rather than *woman.* Yet the words being defined belong to taxonomies other than the ones given by the genus term of their definitions. We need to explore other syntactic cues (such as noun compounding) to membership in multiple taxonomies.

Finally, we extracted a list of nouns ending in the gender-marked suffixes in English, such as *-ess, -ette,* or *-ix,* for [+female] and *-man* for [+male]. The number of endings for [+female] is greater since this is the marked case in English, as seen in cases like *president/ woman president/ *man president* or *lawyer/ woman lawyer/ *man lawyer.* This method accounts for about 15% of the words on the final list.

Results. The lists we have for the features [+male] and [+female] consist of nouns which have the feature in all of its senses. In the original unprocessed lists from sprouting and filtering, there were 519 [+female] nouns, and 732 [+male]. For [+female], there were 230 entries eliminated from the filtered list, 85 added from the key word search, and 43 added from the backwards search for morphologically feminine nouns, giving a final total of 417 nouns marked [+female]. For [+male], there were 540 entries eliminated from the filtered list, and 36 added from the key word search, giving a final total of 228 nouns marked [+male]. Notice that many more male nouns had to be eliminated. The reason for this is that the male case is unmarked in English, and often serves as the generic term referring to both the female and male version of a type. For example, *author* can be used for men or women, whereas *authoress* always refers to women.

The issue of sense discrimination plays a important role in marking words in the lexicon. Many nouns on our lists were *likely* to be either inherently [+male] or inherently [+female]. These words are of two types. The first includes words which have several senses not all of which possess the required feature. For example, among the definitions for *king* in Webster's Seventh are:

king (n)
DEFINITIONS:
1a (often attrib) a male monarch of a major territorial unit . . .
2 (cap) GOD, CHRIST
3 . . . a chief among competitors
4 the principal piece in a set of chessmen
5 a playing card . . .
6 a checker that has been crowned

The last three definitions are clearly inanimate, so marking a chess piece, card, or checker as [+male] is incorrect. However, in a certain sense, the meaning of *king* is intuitively male. Since we have at the present time no way to distinguish senses, we cannot mark certain of the senses [+male] while leaving the other senses neutral. A particular application, however, might have criteria for ignoring certain word senses. For example, in a context dealing with chess, *king* might always bear the features [-human] and [-male]. Most other applications might ignore the [-human] sense of *king.*

The other words which are likely to bear a particular feature are those which are culturally or traditionally associated with a particular attribute. Such words are *prostitute* (a *male prostitute* is usually specified as such), *major-general, Pope* and *beautician.* This information is usually not explicitly stated in a definition. At the moment, we have no mechanism to assign a feature "likely to be X", but this is in our future plans.

Many of the words that appeared on our initial lists were adjectives that are used to refer to nouns which are *likely* to be either [+male] or [+female], such as *petite, husky,* and *platinum-blonde.* If these adjectives were marked as likely to select for nouns of a particular type, then this marking could in turn be used to identify nouns of that type. For example, if an adjective like *petite* were marked with a feature [likely to select for a female noun], then texts could be searched to pick out candidate [+female] nouns based on the presence of [+female] adjectives. This could be the basis for another methodology for identifying nouns possessing a certain semantic or syntactic feature. So far our lexical information is confined to inherent features for nouns but a logical next step is to also mark selectional features for adjectives.

4.1.2. [+UNIT] NOUNS.

We have marked nouns such as *dollar, peso, mile,* and *acre* with the feature [+unit]. The motivation for this arose from the following syntactic facts:

Verb Agreement. It is necessary to "break the rules" of verb agreement when using unit nouns. For example,

(1) Three dollars is too much to pay.
(2)*Three dollars are too much to pay.
(3) Four acres is too large for us.
(4) *Four acres are too large for us.

Agreement within a Noun Phrase. Usually a plural number marker requires a plural noun within the same noun phrase, such as "three dollars" or "four acres." However, within a noun compound, this rule does not apply, even when the noun is clearly plural:

(5) This is a three dollar blouse.
(6)*This is a three dollars blouse.
(7) We have a new dish washer.
(8)*We have a new dishes washer.

There appear to be some differences between unit nouns and other count nouns. It seems to be that unit nouns obligatorily require a verb inflected for single number, whereas other count nouns optionally permit a singular or plural verb as shown in (9)-(10).

(9) Four books is too much to read in one night.
(10) Four books are too much to read in one night.

The methods we used to identify dictionary entries which should be marked with the [+unit] feature are similar to those used for [+male] and [+female]. We obtained two lists, one from sprouting and filtering, and another from the key-word search program.

Results. The number of unit nouns on the original filtered Longman list was 470. The key-word search technique turned up 527 more candidate words. Subtracting the 153 words common to both lists, there remained 844 words to be checked. Half of these passed the criteria in all senses, so a total of 422 words are marked with the [+unit] feature. In addition, a subset of unit nouns (n=128) were marked with a feature [+currency], of which 26 have irregular plural forms, yielding a total of 154 tokens marked. This was an unexpected but useful result particularly for judging the grammaticality of constructions like: Rice is four dollars a pound.

4.1.3. ACTIVE AND STATIVE VERBS.

Markowitz et al. (1986) present linguistic motivations for extracting an "active" and "stative" feature for verbs, and suggest methods for using MRD's as sources for finding active/stative information for verbs. We decided to try to carry out their suggestions using our analysis tools and our morphological analyzer on our dictionary resources. Our attempt was both a success and a failure. The success arises from the design and use of our tools. We achieved clear results, although the results were not what we had expected. The disappointment arises from the fact that the active/stative distinction, it turns out, is not a lexical property at all, but rather a property of sentences. Our tools are geared towards *lexical* properties and are not designed to extract *sentence* or *discourse* properties. It is unclear how or whether to represent such attributes as lexical features.

Linguistic Motivations. The standard division between Stative (or Status) verbs and Process (or active or dynamic) verbs revolves around aspectual constraints. The classic test is the ability to take progressive and imperative aspect as shown in the following examples:

(11) The boy classifies the butterflies.
(12) The boy is classifying the butterflies.
(13) Classify the butterflies!
(14) The boy resembles his father.
(15) *The boy is resembling his father.
(16) *Resemble your father!

The verb *classify* is a process or dynamic verb. The progressive aspect is said to be possible with these verbs because they indicate a condition which can change over time. Similarly, the imperative is possible because one can order someone to do something in order to change the situation or condition. Other examples of dynamic verbs are *ask, grow, hurt, arrive,* and *jump*. Stative verbs, on the other hand, do not indicate a condition which can be changed by an action. As illustrated in (14)-(16), someone either resembles someone else, or he doesn't. Also, it is meaningless to order someone to resemble someone else. "Resemblance" is a condition, and not a changeable state. Process verbs are said to outnumber stative verbs, not only by type but also by token. Quirk and Greenbaum(1972) identify five types of process verbs, but only two types of stative verbs.

Many stative verbs can be used as process verbs with certain changes in meaning. These meaning shifts might be reflected in a dictionary by different sense numbers since the fundamental reason for sense division is semantic. Consider the stative use of the perception verb *hear*:

(18) I hear the music on the radio.
(19) * I am hearing the music on the radio.
(20) The judge is hearing your case first.

As shown in (19) the progressive is ungrammatical when the verb *hear* is used with a stative meaning, whereas (20) is fully grammatical since the interpretation of *hear* is as a process. The problem of identifying both an active and a stative sense of a verb is related to the sense discrimination problem since a shift from active to stative sense has syntactic implications.

Methodology and Tools. First, we obtained the list of noun hyponyms for "act" and "state" from our hyponym file. We then analyzed these nouns using a modified version of the morphological analyzer constrained to give the outermost verb. The expected result was that the outermost verb bases of these nouns would exhibit the active and stative properties depending on the source hypernym. We give some examples of definitions which have the genus *act* or *state*. Observe that *resemblance* which is derived from the stative verb *resemble* has *state* as one of as its genus term; *abbreviation* has *act*; and *abatement* has both *act* and *state*

RESEMBLANCE (n) the quality or state of resembling : . . .

ABBREVIATION (n) the act or result of abbreviating : . . .

ABATEMENT (n) the act or process of abating : the state of being abated . . .

The results of morphological analysis on these sample nouns yield *resemble, abbreviate,* and *abate* as their verb bases.

Our processing of Webster's Seventh showed that there were 2185 words that had the noun *state* as a hypernym, for example, *abatement, abidance, ability,* and *abnormality*. The number of nouns having *act* for the hypernym was just slightly fewer at 2115, for example, *abatement, abbreviation, aberration,* and *abidance*. Not all have verbs bases, e.g. *aberration,* and *abnormality*. Observe that since many nouns have both hypernyms, the corresponding verbs end up on both lists, e.g. for *abatement*. The results are shown in Figure 8.

		Number
(21)	total state nouns	2185
	verb bases	646
(22)	total act nouns	2115
	verb bases	1570
(23)	verbs in state only	265
	verbs in act only	1189
	verbs in both	381

Figure 8. Results of processing "act" and "state" hyponyms.

Once we derived the verbs, we were then ready to test the original hypothesis. If the hypothesis is correct, verbs underlying nouns which have *state* as a hypernym are likely to be stative. Similarly, verbs underlying nouns which have *act* as a hypernym are likely to be active.

We constructed the following syntactic tests from Joos (1968) and Quirk and Greenbaum (1972) which are said to characterize stativity: (1) the simple present progressive, (2) the present progressive with the adverbial modifier "more and more each day" in order to test a continual or process reading, and (3) the imperative. We built a sentence frame generating tool which takes as input each verb to be tested. Test sentences are constructed and the user is asked to respond whether a sentence is acceptable, unacceptable, or questionable. There is also opportunity for comments. The results are placed in a matrix containing the collected judgements for a list of verbs.

Before running our lists of derived verbs through the sentence frame tool, we decided to test it with a verbs which have been said to be undeniably stative and active. We took seventy stative verbs and forty action verbs from Joos (1968) and Quirk and Greenbaum (1972).

To illustrate, given the sample list of three verbs in (24), frame tests were automatically generated for each verb in turn, as shown in (25)-(27). The list in (24) has three verbs: one said to be stative *(resemble)*, one active *(throw)*, and one process *(feel)*. Results are tabulated for each response, and given in (28).

(24) resemble
throw
feel

(25) "RESEMBLE"
TEST 1: He is resembling his mother today.
TEST 2: She is resembling his mother more and more (each day).
TEST 3: Resemble your mother now!

(26) "THROW"
TEST 1: He is throwing the ball today.
TEST 2: She is throwing the ball more and more (each day).
TEST 3: Throw the ball now!

(27) "FEEL"
TEST 1: He is feeling happy today.
TEST 2: She is feeling happier more and more (each day).
TEST 3: Feel happy now!

The table given in (28) shows the results of applying the frame tests to the verbs given in (24): The expected result if the distinction were as it is said to be is given in (29).

(28) Results of Testing Verbs for Stativity:

TEST	1	2	3
resemble	−	+	−
throw	+	−	+
feel.................	+	+	+

(29) Ideal Pattern

TEST	1	2	3
active verb	+	+	+
stative verb.......	−	−	−

Notice that the patterns in (28) do not conform to the expected patterns in (29).

A more complete set of responses using a random subset of the verbs taken from Joos (1968) and Quirk and Greenbaum (1972), again failed to show any clear patterns or any clear distinctions among sets of verbs which are supposedly active and stative.

Despite the fact that textbooks set out explicit distinctions for verb types, there appear to be no significant clear-cut core set of verbs in each category. Rather, there appear to be situations in which a given verb has a stative reading and other situations in which the same verb has an active reading. What is worse, these "situations" can be either semantically or syntactically conditioned.

Choice of Auxiliary Verb in Italian. We next explored the possibility of a correspondence between verbs which tend to be active in English and those which select for the auxiliary *avere* in Italian, and similarly for the stative verbs in English and those requiring *essere*. Grammar books often state that in Romance languages, the choice of auxiliary verb depends on whether the verb is stative or active. We performed two tests of this hypothesis. The first was to extract some verbs from the Collins English-Italian dictionary which were labelled as taking *essere* "to be". Verbs are not marked for *avere* "to have" since it is the unmarked case. We then passed these verbs through a frame test, identical to the one described above (except in Italian, of course.) Results showed that less than 10% of verbs with auxiliary *essere* "to be" pass the test for stativity. We also took the translations of the core set of [+active] and [+stative] verbs from the sample verbs given in Joos(1968) and in Quirk and Greenbaum(1972). We then examined the corresponding Italian verbs taken from the Collins English-Italian dictionary to determine which took *essere* "to be" and *avere* "to have". Results showed that almost all of them took the auxiliary *avere* "to have". Thus, we were unable to correlate *activity* and *stativity* with translation of verbs taking *essere* "to be" and *avere* "to have".

Conclusion. The active-stative distinction appears to be an aspectual feature related to discourse. It is not a lexical feature, although certain verbs occur more naturally whereas others are more tightly constrained. This means that the properties active and stative are qualitatively different from the inherent features [+human] or [+male] for nouns, or from the selectional features [+thatcomp] (takes a *that* complementizer), or [requires a human subject] for verbs. For example, a verb which indicates a true state, such as "resemble", is more likely to occur in a sentence where stative aspect is present, but not necessarily. This being the case, then it is strictly speaking not the job of the lexicographer alone to identify such features. Rather, it is the job of the syntactician and semanticist to identify such contexts, along with the lexicographer who should identify verbs that are potentially active or stative in a given context.

4.2. Synonymy and the Problem of Sense Discrimination.

As described above, sense mapping between dictionaries is one of the most important and challenging problems facing computational lexicography. A related problem is sense disambiguation within a single dictionary. *Webster's Seventh Collegiate Dictionary* (W7) defines *car* as a kind of 'vehicle' but does not indicate that it is sense 4 of *vehicle* which is intended and not senses 1, 2, or 3. This lack of an explicit indication of the hypernym's intended sense limits the usefulness of the hypernym relation as we have currently extracted it because hypernymy is actually a relation between word senses, not a relation between words. As a consequence, our sprouts from hypernyms require human intervention to insure that the proper sense of each node is followed. Without automatic sense disambiguation, taxonomic sprouting is thus relegated to the status of a semi-automatic (rather than a fully automatic) processing tool.

It has been suggested (Lesk(1986)) that sense disambiguation of hypernyms might be achieved by comparing the differentia (the words which appear with the hypernym, e.g., "moving on wheels" in the definition of *car* as a "vehicle moving on wheels") to the words that appear in the definitions of the hypernym's senses. The sense definition which most closely matches the differentia would then be selected as the intended sense of the hypernym. The measure of closeness might be something as simple as the number of content words found in the intersection of words from the differentia and the sense definition. However, to the best of our knowledge, this has not yet been carried out automatically on a large scale. Our disambiguation of synonym senses has followed a somewhat different course based on the assumption that synonymy is a symmetric relation between word senses.

Our procedures for sense disambiguation have been used to process *The Collins Thesaurus* (CT), which is stored as a DAM file with 16,700 keyed records containing a total of 278,000 synonym tokens. Each key is a headword or a boldface run-on word from a main entry. The record associated with each key holds the synonym tokens organized by part-of-speech of the key and, within each part-of-speech, by sense number. Many of the entries on the original tape and the printed version of CT are not marked for part-of-speech, so it was necessary to use UDICT to analyze the parts-of-speech of the synonyms listed under each sense number. From this it could then be inferred under which part-of-speech the sense number belonged (Chodorow and Ravin (1987)).

In a dictionary-style thesaurus such as CT, an entry A may have word B listed as a synonym of its nth sense, and entry B may have word A listed as a synonym of its mth sense. Based on the assumption that synonyms are symmetric, we can mark B in entry A as the mth sense of B, and A in entry B as the nth sense of A. An example of this type of one-to-one mapping in CT is given below.

dense (adj) 1. . . . condensed . . . solid . . . .
2. . . . dull . . . stupid . . .
dull (adj) 1. dense . . . . stupid . . . .
2. . . . callous . . . unsympathetic . . . .
.
.
7. drab . . . muted . . . .

Here, sense 1 of *dull* is synonymous with sense 2 of *dense*. 37% of the 278,000 synonym tokens show this type of symmetry. Of course, there are also mappings of the one-to-many variety, but they account for only .5% of the tokens. We have automatically marked the senses of all synonyms in one-to-one and one-to-many relations.

The third type of mapping, many-to-many, accounts for just .5% of the total, but it poses a problem for the strategy outlined above. This can best be seen by considering an example. Senses 1 and 2 of *institution* list *establishment* as a synonym, and senses 1 and 2 of *establishment* list *institution*. Is sense 1 of *institution* synonymous with sense 1 of *establishment* or with sense 2? The distribution of the terms *institution* and *establishment* alone cannot answer the question, but there may an automatic solution based on the fact that synonyms tend to cluster into groups that represent separate senses. (This phenomenon will be described later in this section.) Consider again the case of *dense* and *dull*. Evidence for linking sense 2 of *dense* with sense 1 of *dull* comes from the symmetric distribution of the two words in the entries. There is however another piece of evidence for linking sense 2 of *dense* with sense 1 of *dull*, and that is the co-occurrence of the word *stupid* in both synonym lists. Thus, taking the intersections of synonym lists might form the basis for an automatic approach to disambiguating the many-to-many mappings. This is similar in some ways to Lesk's suggestion for disambiguating hypernyms by comparing words that appear with the hypernym to words that appear in the hypernym's sense definitions.

Of course, not all tokens are symmetrically mapped; 62% appear in asymmetric relations. There appear to be many reasons for the asymmetry. 20% of the items offered as synonyms are phrases or rare words that simply do not appear as main entries in the thesaurus. The other 42% which do appear as main entries but do not 'point back' are the focus of current analysis. Some of the asymmetries appear to be mere oversights on the part of the lexicographers. For example, *assembly* has *throng* listed as a synonym of one of its senses, but *throng* does not list *assembly* as a synonym, although it does give *assemblage, congregation, multitude,* and other related words. A second type of asymmetry results when a central sense of one word is synonymous with a very peripheral sense of another. One sense of *say* lists *add,* as in "He added that he would like to see the demonstration." The entry for *add* does not however contain this peripheral sense and deals only with the arithmetic *add*. This omission is more serious than simply omitting a word, for here an entire sense seems to have been left out. A third situation that gives rise to asymmetric links is when the relation between the two words is not really synonymy at all but rather hypernymy. For example, *book* lists *manual* as a synonym, but *manual* does not list *book*; instead special types of books such as *handbook* are given.

Once asymmetries have been discovered, it is possible to supply main entries where they are missing (as in the case of rare words or phrases), to insert synonyms into already existing entries, and even to create missing senses of entries. These enhancements to the thesaurus should make it more valuable for computational research and also as a resource for human use.

Other enhancements require a fundamental reorganization of information about synonyms. In a dictionary-style thesaurus, synonymy is treated as a relation between pairs of word senses. Another approach is to consider it a relation among words that share a common sense. It can thus be viewed as a many-to-one mapping between various spellings and a single meaning. The words *dense, dull,* and *stupid* all point to one another because they map to the same sense, although CT, as a dictionary-style thesaurus, has no direct way of indicating this fact.

The many-to-one structure can be seen more explicitly in the traditional Roget's style of organization where a plan of classification clusters words into "ideas" and an alphabetical index of words is used to point to the ideas. In some dictionaries, such as W7, synonym paragraphs list synonyms or nearly synonymous words and describe the subtle, often connotative, differences between them. Main entries for the words point to the paragraphs in much the same way that alphabetically indexed words point to ideas in the traditional thesaurus.

Warnesson (1985) describes how words in a dictionary-style thesaurus can be automatically organized into synonym clusters, each representing a word-sense, and how associations between clusters can be measured. The goal of restructuring the thesaurus in this way is not to reproduce an already existing, externally imposed organizational scheme, such as Roget's, but rather to uncover the implicit structure of the thesaurus based on the actual interconnections among its words. Although we have not yet completed this work, we have developed the tools required for its first three stages.

The first stage in the process is the disambiguation of senses, described above. Next, a set of related words is obtained by sprouting from a selected root node in the synonym index. Because the synonym senses have been disambiguated, no human intervention is required in the sprouting. The third stage is the production of a square matrix in which the rows and columns are labelled with the words obtained from sprouting, and the cells contain either a "1" or a "0" to indicate the presence or absence of a synonym relationship between the row word and the column word. Finally, an algorithm (such as the one reported in Marcotorchino(1986)) is applied to the matrix to produce the clusters.

There are several advantages to organizing synonyms into clusters, as suggested by Warnesson (1985). (1) Space can be saved by having all members of the cluster point to a single location which represents the sense they share, rather than listing in the entry for each member all the other members. (2) The word having the largest number of connections within the cluster and the fewest connections outside it might be offered to the user as the synonym which is most representative of the sense. (3) It is possible to compute the strength of association between clusters based on connections between member and nonmember words. Strong associations should exist between closely related senses, and weak associations between more distant ones. In this way, it might be possible to generate a hierarchical structure of senses from the bottom up, i.e., based on the actual patterns of synonym connections found in the thesaurus. (4) An on-line thesaurus in which sense-disambiguated words have been structured into clusters provides the best of both forms of organization.

5. Applied Computational Lexicology: Bilingual Dictionaries.

5.1. Transfer of Lexical Information Through Bilingual Dictionaries.

The goal of the experiments described in this section is to take information available from a monolingual dictionary for a given language and to use a bilingual dictionary as a transfer device to enable the same information to be transferred to a monolingual dictionary in a second language. The clearest and most successful attempt at this is reported in the first section below on the semantic feature [+human]. Other more speculative attempts with synonyms and with the syntactic feature [+thatcomp] resulted in partial successes and are reported in later sections.

5.1.1. [+HUMAN] Nouns.

Using sprouting and filtering, we extracted a set of English nouns with the feature [+Human] in all senses. Next, we obtained the set of Italian translations for those nouns from the Collins English-Italian dictionary. Finally, we re-translated the Italian nouns back into English, using the Collins Italian-English dictionary.

Our results were very positive. Of the 321 original English words, 296 Italian words were obtained in the forward (E-I) direction; 63 multi-word translations were also found. Re-translating these 296 Italian words in the backward (I-E) direction yielded 373 English words plus 45 English multi-word translations. Examining the two lists of English words showed that out of the 321 original, 157 were not among the final re-translations. Furthermore, out of the 373 re-translations, 167 were new words not in the original.

An analysis of the words obtained gave two clear results. First, only six out of the 296 Italian words obtained in the forward (E-I) direction did not have the feature [+Human] in all of their senses. This feature can therefore be very well transferred (given our constraint on the polysemy of the source data) to the other monolingual dictionary. Second, well over 90% of the 167 new English re-translations were also [+Human]. This demonstrates the effectiveness of this procedure in the task of enlarging the set of words having a given feature.

The procedures used for transferring the [+human] feature from English to Italian can be viewed as instances of a special kind of sprouting called "bilingual sprouting". In this case, the binary relation through which we propagate features is the translation relation, rather than synonymy, hypernymy, or hyponymy. As with other forms of sprouting, we can view the translation relation as transitive, and therefore we are able to propagate lexical information not only from one language to another, but also back to the original language. Of course, again as with normal sprouting, we must take care to control for the effects of polysemy. These considerations lead to the definition of two types of bilingual sprouting: unconstrained and constrained.

Unconstrained bilingual sprouting. This form of bilingual sprouting assumes that if a word has a particular feature in all of its senses, then any translations of that word that are monosemous will also bear that feature. This, of course, excepts grammatical features which only apply in one of the two languages involved. So, for example, we could not propagate grammatical gender from Italian to English.

A reasonable procedure that implements this kind of sprouting would begin with a list of words in the first language (L1) bearing the feature to be propagated. For each word, assign the feature to any of its monosemous translations, given by the bilingual dictionary. This accomplishes the transfer of the feature to the second language (L2). We can continue by inspecting the bilingual dictionary for the other direction (from L2 to L1). Since the words in L2 are all monosemous, any translations from them back into L1 are also likely candidates for the feature.

Despite the care taken by this procedure to deal only with monosemous words in L2, polysemy in L1 and in the bilingual dictionary causes the procedure to yield lists which must be manually checked. However, the labor involved is much less than would be required to do the work entirely by hand. As reported above, when the procedure was used to propagate the [+Human] feature from English to Italian, we obtained a list of Italian words which were correctly marked in much more than 95 percent of the cases. Furthermore, when we reflected the feature back onto English (with the Italian-English dictionary) we obtained a new set of English words not on our original list. More than 90 percent of those were indeed [+Human] words.

Constrained bilingual sprouting. This type of sprouting allows the assignment of features to word senses in L1, rather than to an entire word. It is based on the observation that if a single sense of a word in L1 has a monosemous translation in L2, then both words can be assigned any transferable features that either of them bears. This assumes that the lexicographers who created the dictionaries being used had comparable ideas about what constitutes a word sense. If we can maintain this assumption, then a sprouting procedure based on our observation should run automatically and yield quite reliable results. However, we still imagine that a manual check of the results for validity would be in order.

A further observation is that if the word sense has multiple monosemous translations into L2, then each of the L2 words can receive the union of their transferable features. A possible algorithm based on this observation would use three dictionaries. First, a monolingual dictionary for L1 would contain features to be transferred. Second, a bilingual dictionary from L1 to L2 would serve as the conduit for the lexical information and the pointer to words in L2. Third, a monolingual dictionary for L2 would be the target for the information to be propagated. The algorithm would scan the words on the L1 side of the bilingual dictionary. For any word sense (in the monolingual dictionary for L1) with multiple monosemous translations into L2, we can assign the union of the transferable features (from the L1 word sense and all L2 translations) to to all of the words in the L2 monolingual dictionary.

The two flavors of constrained bilingual sprouting (one for transferring information from L1 to L2, and the other for reflecting it back to L2) would both be strengthened if we could drop the requirement that the words in L2 be monosemous, and map to word senses instead. In the section 5.2.1, we discuss an enhancement to bilingual dictionaries which will have precisely this effect.

Two important points must be made about bilingual sprouting. First, it is only feasible and useful when the lexical information available and desired in the two languages' monolingual dictionaries are the same. As reported above, for Italian and English we have established lists of features and attributes of interest for both languages, and we are confident that these lists match the requirements of prospective users of the monolingual dictionaries. Second, constrained bilingual sprouting obviously requires the prior establishment of the binary mapping relationships between both monolingual dictionaries and the respective sides of the bilingual dictionary.

5.1.2. SYNONYMS OF "SAY".

We have also made a preliminary attempt to transfer a thesaurus entry from the Collins Thesaurus (CT) into Italian by means of the English-Italian and Italian-English bilingual dictionaries. The goal of this effort is to determine the feasibility of using a monolingual English thesaurus and bilingual dictionaries to generate a monolingual Italian thesaurus. The entry selected for the test was the verb *say,* which has six senses listed in CT.

First, the synonyms of each sense of *say* and their synonyms were obtained through sprouting to form a synonym set. Next, each of these synonyms was looked up in the English-Italian dictionary to obtain Italian translations. Of course, many of the translations were inappropriate. As an example, one translation of *add* is *calcolare* ("to calculate"). All of the Italian translations were then looked up in the Italian-English dictionary to obtain their English translations. These were compared to the original set of English synonyms. An Italian word was deleted if none of its English translations was among those in the original synonym set, except of course for the translation that was responsible for its being included in the first place. *Calcolare* was deleted because none of its English translations was found in the original group of English synonyms. In this way, inappropriate Italian words were eliminated from the translation set.

A rectangular matrix was constructed with the columns representing English words and the rows representing Italian words. A "1" was placed at the intersection of a row and a column if the English word and the Italian word were listed as translations in the bilingual dictionary; otherwise the intersection held a "0". An algorithm for clustering binary rectangular matrices (Marcotorchino(1986)) was used to rearrange the rows and columns.

We were particularly interested in seeing if the clusters showed the structure of the original CT entry for *say* with its six senses. The results were somewhat mixed. For very circumscribed senses, such as the sense of *say* related to a performance *(deliver, orate, perform, recite,* etc.), the translations did indeed show the appropriate clustering, but for the more general senses, the structure, if any, was much harder to discern. Although this technique seems to hold some promise, it is too early to say if it is feasible to use an English thesaurus and bilingual dictionaries to automatically produce an Italian thesaurus.

5.1.3. [+THATCOMP] VERBS.

We started with a list of English verbs which were hand-marked with a feature reflecting selection for a *that* complementizer. We constructed a matrix of grammaticality judgments for these verbs with the sentence frame generating program. The four test frame conditions were: (1) VERB obj *that,* (2) VERB ϕ *that,* (3) VERB obj *infinitival complement,* and (4) VERB ϕ. *infinitival complement.* We then took the translations of a test set of these verbs. The same syntactic tests were applied to the Italian verbs. An example of the combined matrix is given below. Each column refers to each syntactic test frame.

TEST	1	2	3	4
acknowledge	+	−	−	+
ammettere	+	−	+	−
add	+	−	−	−
addizionare	−	−	−	−
admit	+	−	−	−
riconoscere	+	−	+	−

The results we present should be viewed as preliminary because we took only a small set of some ten English verbs and some twenty-five Italian verbs. The first observation is that some of the translations do not belong to the same semantic field as the word in L1. An example is *add* and *addizionare*. *Addizionare* means to "to add" in the mathematical sense, but does not mean "to add" in the sense of "comment". When we have lexical information applied to senses, disambiguated references in bilingual dictionaries (see section 5.2.1) and sense mappings (see section 5.2.2) such problems should disappear.

Second, there are the cases of a single entry in L1 with multiple translations in L2. An example of that is *anticipate*:

TEST	1	2	3	4
anticipate	+	−	−	−
aspettarsi	+	−	+	−
precedere	−	−	−	−
pregustare	−	−	−	−
prevedere	+	−	+	−
prevenire	−	+	−	−

There is a possibility of having a *that* clause for at least one of the translations in each group of translations. Further, it appears that the pattern could serve as a clue to pick which of the many translations is the one that corresponds semantically to the English word sense taking *that*. This applies to the first two test frames, i.e. columns one and two in the table. For example in the case of *anticipate, aspettarsi* and *prevedere* are the two translations with the meaning closest to *anticipate* when *anticipate* is used in the sense taking a *that* clause. Our conclusion is that this process is not feasible completely automatically, but that such a procedure could reduce much searching and guessing. Further, we believe we could profit by utilizing information on *that* clauses found in example sentences in the bilingual dictionary. This could provide an additional mechanism to eliminate non-corresponding translations.

5.2. Cross-References and Sense Disambiguation in Bilingual Dictionaries.

Since the Collins bilingual dictionaries play such an important role in our research, both as a conduit for lexical information transfer and as reference material for researchers, it is desirable to make them as accurate and useful as possible. This section discusses two methods of enhancing the contents of those bilingual dictionaries. The first method supplies sense numbers to the cross-references between the two sides of the dictionary. The second supplies "missing information" in the form of cross-references which should be there but are not.

5.2.1. Disambiguating References in Collins Bilingual Dictionaries.

The English-Italian side of Collins assigns sense numbers to the English words. Similarly, the Italian-English side assigns sense numbers to the Italian words. A desirable enhancement to the entries in either side is to assign sense numbers to the translations so that they accurately point to the other side. Pictorially, we want to instantiate sense references represented by the arrows in Figure 9. This result is essentially equivalent to that discussed in section 4.2 for disambiguating the cross-references in the Collins Thesaurus (CT).

In terms of LDBs, we are proposing that the structure of the current translation attribute "xlat" be enhanced by the addition of a new "senseno" (sub)attribute identifying the sense number on the other side of the dictionary for the word given as the value of the "spel" (sub)attribute. The resulting "xlat" attribute will look like:

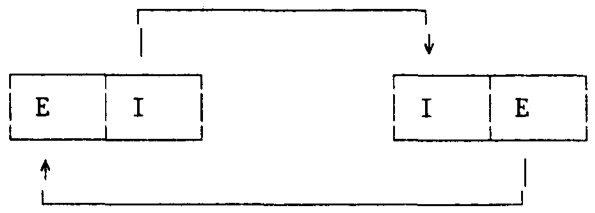

Figure 9. Disambiguating cross-references in a bilingual dictionary.

```
...
|
+-xlat
  |
  +-spel
  +-senseno
  +- ...
```

As with the mapping files, several procedures will be required in order to determine the correct values for these new "senseno" attributes. Some of these procedures will be trivial, as when the words involved are monosemous. Recall that, in the case of sense mapping with Webster's Seventh, the majority of words are monosemous. We expect to find the same situation in Collins. A version of the procedure given in Section 4.2 should supply many values. Specifically, if any sense of an English word, E1, gives a particular Italian word, I1, as its translation, and if one sense of I1 gives E1 as its translation, then we can mark the I1 translation of E1 (on the English-Italian side) with the sense number of I1 (on the Italian-English side) that had the reciprocal reference. Further, more difficult, and possibly even manual procedures may be required to obtain the rest. In any event, once the sense links have been established, they will remain permanently valid, as is the case for the inter-dictionary mapping files.

An immediate benefit that would emerge from disambiguating the references in a bilingual dictionary would be that we can increase the power of constrained bilingual sprouting. We would then be able to map directly onto word senses of polysemous words in the second language and would not be restricted to using only monosemous words. The reader is invited to refer back to the discussion of bilingual sprouting, in section 5.1.1, for details.

5.2.2. SYMMETRIZING COLLINS BILINGUAL DICTIONARIES.

Inspection of the Collins English-Italian and Italian-English dictionaries reveals that there are many asymmetries between the two sides. An example of an asymmetry would be a case where an English word gives an Italian word as its translation, but that Italian word does not, in turn, give the original English word as its translation. In order to provide improved access to bilingual information by humans and by our analysis procedures, it seems feasible to produce a symmetrized bilingual dictionary. Essentially, this would involve automatically locating all asymmetries and adding additional entries and/or translations to make them symmetric. It should not be difficult to accomplish this task, given LDBs for the two dictionaries.

Despite its initial appeal, we suspect that the creation of a fully symmetrized bilingual dictionary may not be totally desirable. It may be that some of the asymmetries exist because of valid lexicographic principles, and that violation of those principles will lead to an undesirable result. Through inspection of the dictionary we have pinpointed four candidate principles which could warrant the use of asymmetric references.

1. Lemma forms of words which are only (or mostly) used in derived forms should not be given as translations of words in the other language. Thus, although *allege* appears in the English-Italian dictionary, it is never given as the translation of an Italian word. This may be because *allege* is almost always used in English in its past participial form: *alleged*. In the corpus analyzed in Kucera and Francis (1967), *alleged* occurs ten times while *allege* occurs only once.
2. Rare words or words referring to highly specific concepts may be translated with more general words, however a general word should not be translated with a specific or rare one. Thus, English *moment* is translated as Italian *importanza* in its sense of "importance". However, since the "importance" meaning of English *moment* is quite specific, it is not given as the translation of Italian *importanza*. This case seems similar to the example of *say* and *add* given in the discussion of asymmetry in CT. In the case of bilingual dictionaries, perhaps the principle is to avoid offering such rare word senses to the non-native speaker, who would find them difficult to use correctly.
3. In apparent violation of the preceding principle, we sometimes find a general word translated with what are essentially hyponyms in the other language. Thus, for *book,* Collins English-Italian gives *quaderno* "(notebook)", *bustina* "(of matches)", and *blocchetto* "(of tickets)". The Italian-English side of Collins only gives specific translations for these words; none is translated as *book*. This seems entirely equivalent to the case of *book* and *manual,* covered in the discussion of CT. The principle here may be to avoid giving as translations either hypernyms or generic words which acquire a particular meaning only in a specific context. In the case of "book", the hypernym would be completely redundant with respect to the specific translations that are given and would thus add no new information.
4. Some translations are not symmetric because the translation given does not appear as an entry on the other side of the dictionary. For example, Collins lists *to parse* as one of the English translations for Italian *analizzare*. However the English-Italian side of Collins does not contain the word *parse*. Similarly, for Italian *codifica,* Collins gives *codification* and for Italian *proclamare* it gives *promulgate*; neither of these English words appears as a head word on the English-Italian side of the dictionary. These omissions may merely reflect an oversight on the part of the lexicographers.

We assume that principles whose only purpose is to save space can be safely violated by the creation of the missing symmetric links. Our approach to sorting out the question of whether symmetrizing is a good thing in general is to generate candidate links and to see if they violate these or other principles and whether we like the result. We offer the preceding list of types of asymmetry not as hypotheses to be verified, but merely as things to look for during such an analysis. In the process, we may discover the real principles which favor the creation of asymmetric links, if any exist. The best outcome may be that we can build a partially symmetrized dictionary containing all desirable symmetry and no undesirable asymmetry. A further benefit is that we can provide a tool which allows the lexicographers to inspect the asymmetries in their dictionaries before committing them to print.

6. CONCLUSION.

We have outlined a research program intended to provide a comprehensive set of tools and methods for using machine readable dictionaries to produce computerized dictionaries suitable for use in natural language processing systems. We have presented operational tools and analysis results which we have obtained with them. A wide variety of techniques from computer science, linguistics, and lexicography need to be combined in order to succeed at building the dictionaries we envision. The project described here provides a coherent framework and computational basis for proceeding.

ACKNOWLEDGEMENTS.

We are grateful to Yael Ravin and Howard Sachar for their major and continuing contributions to the analysis of the Collins Thesaurus. Barbara Kipfer performed significant early analyses of semantic features, using sprouting and filtering. Gustaf Neumann and Seved Andersson added important new functions to the Dictionary Access Method, and Neumann used the new DAM to build WordSmith functions for Webster's Seventh and Longman. We thank the group working on the Italian Machine Readable Dictionary at the Institute for Computational Linguistics in Pisa for ideas conveyed in the present work. We also thank Robert Ingria for suggesting the use of direct nominalizations in the study of active and stative verbs.

REFERENCES.

Alshawi, H. 1985 Processing Dictionary Definitions with Phrasal Pattern Hierarchies. Unpublished paper. University of Cambridge Computer Laboratory, Cambridge, England.

Amsler, R. A. 1980 The Structure of the Merriam-Webster Pocket Dictionary. Doctoral Dissertation, TR-164, University of Texas, Austin.

Byrd, R. J. 1983 Word Formation in Natural Language Processing Systems. *Proceedings of IJCAI-VIII*:704-706.

Byrd, R. J. 1986a Dictionary Systems for Office Practice. IBM Research Report RC 11872, T.J. Watson Research Center, Yorktown Heights, New York.

Byrd, R. J. 1986b The TUPLES Text Analysis System. IBM Research Report, T.J. Watson Research Center, Yorktown Heights, New York.

Byrd, R. J. and M. S. Chodorow. 1985 Using an On-line Dictionary to Find Rhyming Words and Pronunciations for Unknown Words. *Proceedings of the Association for Computational Linguistics*: 277-283.

Byrd, R. J., J. L. Klavans, M. Aronoff, and F. Anshen. 1986a Computer Methods for Morphological Analysis. *Proceedings of the Association for Computational Linguistics*: 120-127.

Byrd, R. J., G. Neumann, and K. S. B. Andersson. 1986b. DAM— A Dictionary Access Method. IBM Research Report, T.J. Watson Research Center, Yorktown Heights, New York.

Calzolari, N. 1980 Polisemia e Omografia nel Dizionario Macchina dell'Italiano. In: *Studi di Lessicografia Italiana*. Accademia della Crusca, Firenze: Vol. II:283-313.

Calzolari, N. 1983 Semantic Links and the Dictionary. In: Burton, S. K. and D.D.Short. eds., *Proceedings of the Sixth International Conference on Computers and the Humanities*. Computer Science Press, Rockville, Maryland: 47-50.

Calzolari, N. 1984a Detecting Patterns in a Lexical Data Base. *Proceedings of COLING84*. 170-173.

Calzolari, N. 1984b Machine-readable Dictionaries, Lexical Data Bases, and the Lexical System. *Proceedings of COLING 84*. 460.

Chodorow, M. S., R. J. Byrd, and G. E. Heidorn. 1985 Extracting Semantic Hierarchies from a Large On-line Dictionary. *Proceedings of the Association for Computational Linguistics*: 299-304.

Chodorow, M. S. and Y. Ravin. 1987 (in preparation) Analyzing the Collins Thesaurus. IBM Research Report, T.J. Watson Research Center, Yorktown Heights, New York.

Collins. 1971. Collins *Spanish Dictionary: Spanish-English. English-Spanish*. Collins Publishers, Glasgow.

Collins. 1978 *Collins Robert French Dictionary: French-English. English-French*. Collins Publishers, Glasgow.

Collins. 1980 *Collins German Dictionary: German-English. English-German*. Collins Publishers, Glasgow.

Collins. 1980 *Collins Sansoni Italian Dictionary: Italian-English. English-Italian*. Collins Publishers, Glasgow.

Collins. 1984 *The New Collins Thesaurus*. Collins Publishers, Glasgow.

Dahlgren, K., and J. McDowell. 1986 Kind Types in Knowledge Representation. *Proceedings of COLING86*. Bonn, Germany.

Heidorn, G. E., K. Jensen, L. A. Miller, R. J. Byrd, and M. S. Chodorow. 1982 The EPISTLE Text-Critiquing System. *IBM Systems Journal* 21:305-326.

IBM. 1978 *Query-by-Example: Terminal Users Guide*. IBM form no. SH20-2078.

Ingria. R. J. P. 1986 Structuring The Lexicon. Tutorial Session Paper presented at the 24th Annual Meeting of the Association for Computational Linguistics. Columbia University, New York, New York.

Jensen, K., G.E. Heidorn, L.A. Miller, and Y. Ravin. 1983 Parse Fitting and Prose Fixing: Getting a Hold on Ill-formedness. *AJCL* 9(3-4):123-36.

Joos, M. 1968 *The English Verb: Form and Meanings*. University of Wisconsin Press, Madison, Wisconsin.

Kipfer, B. A. 1985 Marking Semantic Features to Enhance a Dictionary for Natural Language Processing Applications. Lexical Systems Project Report, T.J. Watson Research Center, Yorktown Heights, New York.

Kucera, H. and W. N. Francis. 1967 *Computational Analysis of Present-Day American English*. Brown University Press, Providence, Rhode Island.

Langendoen, D. T., and H. M. Barnett. 1987 (to appear) PLNLP: A Linguist's Introduction. IBM Research Report. T.J. Watson Research Center, Yorktown Heights, New York.

Lesk, M. 1986 Automatic Sense Disambiguation Using Machine-readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. *Proceedings of 1986 SIGDOC Conference*.

Longman. 1978 *Longman Dictionary of Contemporary English*. Longman Group, London.

Marcotorchino, F. 1986 Maximal Association Theory. In: *Classification as a Tool for Research*. W. Gaul and M. Schader, eds., North Holland.

Markowitz, J., T. Ahlswede, M. Evans. 1986 Semantically Significant Patterns in Dictionary Definitions. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*: 112-119.

McCord, M. C. 1987 Natural Language Processing in Prolog. In: Walker. A., McCord, M., Sowa. J., and Wilson, W., *Knowledge Systems and Prolog*. Addison-Wesley, Waltham, Massachusetts.

Merriam. 1963 *Webster's Seventh New Collegiate Dictionary* G. & C. Merriam, Springfield, Massachusetts.

Michiels, A. 1982 *Exploiting a Large Dictionary Data Base*. PhD. Dissertation, University of Liege, Liege, Holland.

Neff, M. S. and R. J. Byrd. 1987 WordSmith Users Guide. IBM Research Report. T.J. Watson Research Center, Yorktown Heights, New York.

Neff, M. S., R. J. Byrd, and O. A. Rizk. 1988 Creating and Querying Lexical Data Bases. *Proceedings of the Association for Computational Linguistics Second Applied Conference on Natural Language Processing*: 84-92.

Pereira, F. and D. Warren. 1980 Definite Clause Grammars for Language Analysis — a Survey of the Formalism and a Comparison with Augmented Transition Networks. *Artificial Intelligence* 13:231-178.

Quirk, R., S. Greenbaum, G. Leech, and J. Svartvik. 1972 *A Grammar of Contemporary English*. Longman, London.

Tompa, F. 1986 (unpublished) Database Design for a Dictionary of the Future. University of Waterloo, Waterloo, Canada.

Warnesson, I. 1985 Optimization of Semantic Relations by Data Aggregation Techniques. *Journal of Applied Stochastic Models and Data Analysis*, 1(2).